Designing for Engagement: Building IT Systems

Mr Malcolm Corney
Faculty of Science and Technology
m.corney@qut.edu.au

Keywords: design, introductory computer programming, information technology systems

Context

The School of Information Technology at QUT has recently undertaken a major restructuring of their Bachelor of Information Technology (BIT) course. Some of the aims of this restructuring include a reduction in first year attrition and to provide an attractive degree course that meets both student and industry expectations. Emphasis has been placed on the first semester in the context of retaining students by introducing a set of four units that complement one another and provide introductory material on technology, programming and related skills, and generic skills that will aid the students throughout their undergraduate course and in their careers.

This discussion relates to one of these four fist semester units, namely Building IT Systems. The aim of this unit is to create small Information Technology (IT) systems that use programming or scripting, databases as either standalone applications or web applications. In the prior history of teaching introductory computer programming at QUT, programming has been taught as a stand alone subject and integration of computer applications with other systems such as databases and networks was not undertaken until students had been given a thorough grounding in those topics as well. Feedback has indicated that students do not believe that working with a database requires programming skills. In fact, the teaching of the building blocks of computer applications have been compartmentalized and taught in isolation from each other.

The teaching of introductory computer programming has been an industry requirement of IT degree courses as many jobs require at least some knowledge of the topic. Yet, computer programming is not a skill that all students have equal capabilities of learning (Bruce et al., 2004) and this is clearly shown by the volume of publications dedicated to this topic in the literature over a broad period of time (Eckerdal & Berglund, 2005; Mayer, 1981; Winslow, 1996).

The teaching of this introductory material has been done pretty much the same way over the past thirty years. During this period of time that introductory computer programming courses have been taught at QUT, a number of different programming languages and programming paradigms have been used and different approaches to teaching and learning have been attempted in an effort to find the golden thread that would allow students to learn this complex topic. Unfortunately, computer programming is not a skill that can be learnt in one semester. Some basics can be learnt but it can take many years to master (Norvig, 2001).

Faculty data typically has shown a bimodal distribution of results for students undertaking introductory programming courses with a high proportion of students receiving a high mark and a high proportion of students receiving a low or failing mark. This indicates that there are students who understand and excel with the introductory material while there is another

group who struggle to understand the concepts and practices required to be able to translate a specification or problem statement into a computer program that achieves what is being requested.

The consequence of a large group of students failing the introductory programming course has been a high level of attrition amongst first year students. This attrition level does not provide good continuity in student numbers in later years of the degree program and the current approach is not seen as sustainable.

Action taken

The design goal for the core of the new BIT degree is to improve student engagement, and consequently progression, while maintaining the quality of graduates. The core units provide a common set of skills and knowledge for graduates from the degree.

The Building IT Systems unit has been designed to be an interactive, interesting and inspirational introduction to how IT applications and systems work. While there is a need for formal lecture material on the main topics, the majority of the learning will be accomplished with a hands-on approach in laboratory sessions. The students will be introduced to the building blocks of larger systems including programming and scripting, database creation and use, and World Wide Web development.

Combining programming, database and web development into one first year unit should allow the students to gain an earlier understanding of these basic concepts albeit at a more general level. This is intended to engage the students in these building blocks so that they can learn the basics by being involved with a variety of interesting practical tasks that will use one, two or all three of the technologies.

Tasks will be developed by groups but group work *per se* will not be assessed. Tasks will be worked on in practical sessions and the students will use a problem solving framework to provide a scaffold for the learning in the unit. Students will be expected to take an inventory of their current skills at the beginning of the task and to determine an approach that will complete the task.

The tasks will be somewhat open ended in their definition, allowing those students who have pre-existing skills to use those skills, while students who have do not have prior knowledge can benefit from supporting material which will be given in lectures and readings and by doing example activities. After a task has been completed, students will be required to reflect on

The assessment for this unit will be an individual portfolio of the activities that each student has carried out during the semester along with reflections on the learning that they have achieved from each of those activities. The portfolio will be submitted twice, allowing the students to receive feedback on their submission before finally being graded at the end of semester.

Tips and tricks

What is seen as essential in achieving success in this unit, and the other first semester units, is for all units in the semester to have a common philosophy of providing the best outcomes for the students involved. This will require a great deal of dedication to these units from the academic co-ordinators and from all teaching staff involved.

Because the units are quite large, more than half of the tutorials and practicals will be led by sessional academics. There are plans in place to provide these vital front line teachers with training and support to be able to deliver the common philosophy of the course.

Results, evaluation, impact

The unit discussed here and the other units in the first semester of the new Bachelor of IT course will be delivered for the first time in 2009, so there are currently no results or evaluation of how the approach has fared.

The Building IT Systems unit will by default be evaluated by the University's normal unit evaluation instrument, the Learning Experience Survey. It is also planned to engage a focus group at the end of semester to gain more meaningful and direct feedback.

Further resources

Nil

References

Bruce, C., Buckingham, L., Hynd, J., McMahon, C., Roggenkamp, M., & Stoodley, I. (2004). Ways of Experiencing the Act of Learning to Program: A Phenomenographic Study of Introductory Programming Students at University. *Journal of Information Technology Education, 3*.

Eckerdal, A., & Berglund, A. (2005). *What Does It Take to Learn 'Programming Thinking'?* Paper presented at the The First International Computing Education Research Workshop, Seattle, Washington, USA.

Mayer, R. E. (1981). The Psychology of How Novices Learn Computer Programming. *ACM Computing Surveys, 13*(1), 21.

Norvig, P. (2001). Teach Yourself Programming in Ten Years. Retrieved 20th January, 2009, from http://norvig.com/21-days.html

Winslow, L. E. (1996). Programming Pedagogy - A Psychological Overview. *ACM SIGCSE Bulletin, 28*(3), 6.

Contact details (details for primary author only)

Postal Address (work):
Email: m.corney@qut.edu.au
Phone: (07) 3138 1923
Fax: (07) 3138 1801